UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P O Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
10/840,164	05/06/2004	Nicolai Kosche	188378/US/2	7512

66083   7590   03/03/2009
SUN MICROSYSTEMS, INC.
c/o Dorsey & Whitney LLP
370 SEVENTEENTH ST.
SUITE 4700
DENVER, CO 80202

EXAMINER
TECKLU, ISAAC TUKU

ART UNIT	PAPER NUMBER
2192	

MAIL DATE	DELIVERY MODE
03/03/2009	PAPER

Please find below and/or attached an Office communication concerning this application or proceeding.

The time period for reply, if any, is set in the attached communication.

PTOL-90A (Rev. 04/07)

	Application No.	Applicant(s)
	10/840,164	KOSCHE ET AL.
Office Action Summary	Examiner	Art Unit	
	ISAAC T. TECKLU	2192	

-- The MAILING DATE of this communication appears on the cover sheet with the correspondence address --

Period for Reply

A SHORTENED STATUTORY PERIOD FOR REPLY IS SET TO EXPIRE <u>3</u> MONTH(S) OR THIRTY (30) DAYS, WHICHEVER IS LONGER, FROM THE MAILING DATE OF THIS COMMUNICATION.
- Extensions of time may be available under the provisions of 37 CFR 1.136(a). In no event, however, may a reply be timely filed after SIX (6) MONTHS from the mailing date of this communication.
- If NO period for reply is specified above, the maximum statutory period will apply and will expire SIX (6) MONTHS from the mailing date of this communication.
- Failure to reply within the set or extended period for reply will, by statute, cause the application to become ABANDONED (35 U.S.C. § 133).
  Any reply received by the Office later than three months after the mailing date of this communication, even if timely filed, may reduce any earned patent term adjustment. See 37 CFR 1.704(b).

Status

1)☒ Responsive to communication(s) filed on <u>13 January 2008</u>.

2a)☐ This action is **FINAL**.   2b)☒ This action is non-final.

3)☐ Since this application is in condition for allowance except for formal matters, prosecution as to the merits is closed in accordance with the practice under *Ex parte Quayle*, 1935 C.D. 11, 453 O.G. 213.

Disposition of Claims

4)☒ Claim(s) <u>1-3,5-12,14-27,29-37,39-42,44-50,52-56 and 58</u> is/are pending in the application.

    4a) Of the above claim(s) _____ is/are withdrawn from consideration.

5)☐ Claim(s) _____ is/are allowed.

6)☒ Claim(s) <u>1-3,5-12,14-27,29-37,39-42,44-50,52-56 and 58</u> is/are rejected.

7)☐ Claim(s) _____ is/are objected to.

8)☐ Claim(s) _____ are subject to restriction and/or election requirement.

Application Papers

9)☐ The specification is objected to by the Examiner.

10)☐ The drawing(s) filed on _____ is/are: a)☐ accepted or b)☐ objected to by the Examiner.

    Applicant may not request that any objection to the drawing(s) be held in abeyance. See 37 CFR 1.85(a).

    Replacement drawing sheet(s) including the correction is required if the drawing(s) is objected to. See 37 CFR 1.121(d).

11)☐ The oath or declaration is objected to by the Examiner. Note the attached Office Action or form PTO-152.

Priority under 35 U.S.C. § 119

12)☐ Acknowledgment is made of a claim for foreign priority under 35 U.S.C. § 119(a)-(d) or (f).

    a)☐ All b)☐ Some * c)☐ None of:

      1.☐ Certified copies of the priority documents have been received.

      2.☐ Certified copies of the priority documents have been received in Application No. _____.

      3.☐ Copies of the certified copies of the priority documents have been received in this National Stage application from the International Bureau (PCT Rule 17.2(a)).

    * See the attached detailed Office action for a list of the certified copies not received.

Attachment(s)

1)☐ Notice of References Cited (PTO-892)

2)☐ Notice of Draftsperson's Patent Drawing Review (PTO-948)

3)☐ Information Disclosure Statement(s) (PTO/SB/08)
    Paper No(s)/Mail Date _____.

4)☐ Interview Summary (PTO-413)
    Paper No(s)/Mail Date. _____.

5)☐ Notice of Informal Patent Application

6)☐ Other: _____.

DETAILED ACTION

1.   Claims 4, 13, 28, 38, 43, 51 and 57 have been cancelled.

2.   Claims 1-3, 5-12, 14-27, 29-37, 39-42, 44-50, 52-56 and 58 have been examined.

Continued Examination Under 37 CFR 1.114

3.   A request for continued examination under 37 CFR 1.114, including the fee set forth in

37 CFR 1.17(e), was filed in this application after final rejection. Since this application is

eligible for continued examination under 37 CFR 1.114, and the fee set forth in 37 CFR 1.17(e)

has been timely paid, the finality of the previous Office action has been withdrawn pursuant to

37 CFR 1.114. Applicant's submission filed on 01/13/2008 has been entered.

Specification

4.   The disclosure is objected to because it contains an embedded hyperlink

http://h30097.www3.hp.com/dcpi/ on page 2, paragraph [1007]. Applicant is required to delete

the embedded hyperlink and/or other form of browser-executable code. See MPEP § 608.01.

Response to Arguments

5.   Applicant's arguments filed 01/13/2008 have been fully considered but they are not

persuasive.

   Argument:

   The Applicant argued that "Yates fails to teach or suggest that the location within the

computer system at which program is being hindered is a virtual memory location as required by

claim 1." (Remark, pages 9-10).

   Response:

      In response to the above argument, the examiner would like to indicate that it is known

   that when a program is executed on a computer, the program refers to memory by virtual

   address. Thus, profile information is recorded describing memory references made by the

   program, the profile information recording physical addresses of the profiled memory

   references. The prior art of record Yates teaches the concept of virtual memory address among

   others in the following columns.

   "Particular embodiments of the invention may include one or more of the
   following features. The regions may be pages managed by a virtual memory
   manager. The indications may be stored in a <u>virtual address translation entry,</u>
   in a table whose entries are associated with corresponding virtual pages, in a
   table whose entries are associated with corresponding physical page frames, in
   entries of a translation look-aside buffer, or in lines of an instruction
   cache. The code at the first destination may receive floating-point arguments
   and return floating-point return values using a register-based calling
   convention, while the code at the second destination receives floating-point
   arguments using a memory-based stack calling convention, and returns
   floating-point values using a register indicated by a top-of-stack pointer." (emphasis added – col. 4:45-55).

   "Tapestry processor 100 fetches (stage 110) instructions from instruction
   cache (I-cache) 112, or from memory 118, from a location specified by IP

(instruction pointer, generally known as the PC or program counter in other
machines) 114, with virtual-to-physical address translation provided by I-TLB
(instruction translation look-aside buffer) 116. The instructions fetched from
I-cache 112 are executed by a RISC execution pipeline 120. In addition to the
services provided by a conventional I-TLB, I-TLB 116 stores several bits 182,
186 that choose an instruction environment in which to interpret the fetched
instruction bytes. One bit 182 selects an instruction set architecture (ISA)
for the instructions on a memory page. Thus, the Tapestry hardware can readily
execute either native instructions or the instructions of the Intel X86 ISA.
This feature is discussed in more detail in section II, infra." (emphasis added – col.22:45-55).

"At the next X86 instruction boundary 566, the information from the
just-completed instruction is clocked from signals 558, 559, 561 to registers
568, 569, 570. Registers 568, 569, 570 are simply a buffer for time-shifting
information about an X86 instruction to make it available during the next
instruction, in case a profile event is to be captured. Because the native
control transfer instruction is always the last instruction of the recipe for
an X86 transfer instruction, the virtual-to-physical translation of the address
of the destination of the transfer (especially in the case of a TLB miss) is
not available until the transfer instruction itself is complete." (emphasis added – col.69:40-55).

From the above cited portions, it is clear that Yates teaches fetching instruction from the

memory by tracking the virtual memory address and translates the memory address to physical

address. Thus, it is respectfully submitted that the above argument is not persuasive and

accordingly the rejection has been maintained as set forth in the Office Action.

Claim Rejections - 35 USC § 102

6.   The following is a quotation of the appropriate paragraphs of 35 U.S.C. 102 that form the

basis for the rejections under this section made in this Office action:

A person shall be entitled to a patent unless –

(e) the invention was described in (1) an application for patent, published under section 122(b), by another

filed in the United States before the invention by the applicant for patent or (2) a patent granted on an

application for patent by another filed in the United States before the invention by the applicant for patent,

except that an international application filed under the treaty defined in section 351(a) shall have the effects

for purposes of this subsection of an application filed in the United States only if the international

application designated the United States and was published under Article 21(2) of such treaty in the English

language.

7.   Claims 1-3, 5-12, 14-27, 29-37, 39-42, 44-50, 52-56 and 58 are rejected under 35

U.S.C. 102(e) as being anticipated by Yates, Jr. et al. (US 7,111,290 B1), hereinafter Yates.

        As per claim 1(Currently Amended), Yates discloses a <u>tangible computer readable</u>

<u>storage medium implemented</u> software tool that determines at least one data address from one or

more instruction instances (col. 2:15-25 "… figures out the address …" and col. 7:15-40 "…

execution speed … instance of instruction have an event code that leaves intact an event code

previously determined…" and see at least col.6:60-67 and col.7:1-10), and that identifies one or

more memory reference objects, <u>which are</u> associated with the data address, as hindering

execution of code that includes the instruction instances, wherein the instruction instances

correspond to the code execution hindrance<u>, wherein the memory reference objects include</u>

<u>virtually addressable memory</u> (col. 6:51-59 "… profile information is recorded that records

physical memory reference …" and col. 7:15-40 "… execution speed … instance of instruction

have an event code that leaves intact an event code previously determined …", col. 4:45-55,

col.17:30-50 col.22:45-55 and col.69:40-55 – virtual address-).

        As per claim 2, Yates discloses the software tool of claim 1 wherein the memory

reference objects include one or more of physical memory reference objects and logical memory

reference objects (col. 7:14-25 "… physical memory reference … reference may record the event

of a sequential execution flow…" and col. 17:15-20 "… memory references referring to logical

address …).

As per claim 3, Yates discloses the software tool of claim 2 wherein the physical memory

reference objects include one or more of cache (e.g. Fig. 1a, 112 and related text), cache lines

(col.88: 15-20 ".. cache lines…"), cache levels (e.g. Fig. 1c, DATA CACHE and related text),

cache sub-blocks (e.g. Fig. 1c, 112 and related text), memory controllers (e.g. Fig. 1a, 124 and

memory modification monitor and related text), addressable memory (e.g. Fig. 1c, 146 and

related text), and memory-management page translation units (e.g. Fig. 1a, 170 and related text).

As per claim 5, Yates discloses the software tool of claim 2 wherein the logical memory

reference objects include one or more of source-level data objects, memory segments (e.g. Fig.

1D and related text), heap variables (col. 30:40-50 "heap"), variable instances (col. 30:40-50

"state variable"), and stack variables (col. 30:40-50 "stack ").

As per claim 6, Yates discloses the software tool of claim 5 wherein the source-level data

objects include one or more of functions (e.g. Fig. 1e and related text), statically linked objects

(e.g. Fig. 1e, 56 and related text), data structures (e.g. Fig.1e, 48 and related text), data types (e.g.

Fig. 1e, 40 and related text), data type definitions (e.g. Fig. 1e, 32 and related text), operands

(e.g. Fig. 3g, 317 and related text), and expressions (e.g. Fig. 3g, 317 and related text).

As per claim 7, Yates discloses the software tool of claim 6 wherein the statically linked

objects include one or more of global variables and static variables (e.g. TABLE 1 and related text).

      As per 8, Yates discloses the software tool of claim 1 wherein the software tool includes one or more of a compiler, an interpreter (col. 19:20-40 "... interpreter..."), an optimization tool (col.19:20-30 "... emulator..."), and a virtual machine (e.g. Fig. 1a, 118 and related text).

      As per claim 9, Yates discloses the software tool of claim 1 wherein the code includes one or more of machine code (e.g. Fig. 1a, 118 and related text), byte code (col.19:20-30 "... emulator..."), and interpreted code (col. 19:20-40 "... interpreter...").

      As per claim 10, Yates discloses the software tool of claim 1 that also aggregates addresses based on the memory reference objects (e.g. Fig. 1c and related text).

      As per claim 11, Yates discloses the software tool of claim 10 wherein the software tool utilizes at least a portion of the data addresses to aggregate the addresses (e.g. Fig. 1c and related text).

      As per claim 12, Yates discloses the software tool of claim 10 that also provides the aggregated addresses and an indication of the code execution hindrance corresponding to the aggregated addresses for one or more of storage and display (e.g. Fig. 1c and related text)..

As per claim 14, Yates discloses the software tool of claim 1 wherein the code execution hindrance corresponds to one or more sampled runtime events (e.g. Fig. 1d and related text)..

As per claim 15, Yates discloses the software tool of claim 14 wherein the sampled runtime events include one or more of cache misses, cache references, data translation buffer misses, data translation buffer references, and counter condition events (e.g. Fig. 1b and related text).

As per claim 16 (Currently Amended), Yates discloses a method for profiling code executing in a computer system, the method comprising:

identifying an instruction instance that corresponds to a runtime event col. 6:51-59 "… profile information is recorded that records physical memory reference …" and col. 7:15-40 "… execution speed … instance of instruction have an event code that leaves intact an event code previously determined …");

determining a data address from the instruction instance (col. 2:15-25 "… figures out the address …"); and

determining a memory reference object from the determined address (col. 7:14-25 "… physical memory reference … reference may record the event of a sequential execution flow…" and col. 17:15-20 "… memory references referring to logical address …),wherein the data address includes a virtual address in the computer system see also, col. 4:45-55, col.17:30-50 col.22:45-55 and col.69:40-55).

As per claim 17, Yates discloses the method of claim 16 wherein the runtime event is a

sampled runtime event (e.g. Fig. 6c, 650 and related text).

As per claim 18, Yates discloses the method of claim 16 wherein identifying the

instruction

instance comprises backtracking from a second instruction instance to the instruction instance

(e.g. Fig. 650 and related text).

As per claim 19, Yates discloses the method of claim 16 wherein determining the address

from the instruction instance comprises decoding the instruction instance (e.g. Fig. 5b, 556 and

related text).

20. The method of claim 19 further comprising:

decoding the instruction instance if a register that hosts the instruction instance

is determined as valid (e.g. Fig. 5b, 556 and related text).

As per claim 21, Yates discloses the method of claim 20 wherein determining if the

register is valid comprises:

applying reverse register transformation with respect to the runtime event (e.g. Fig. 3d

and related text); and

determining whether the register is valid based on the applied reverse register

transformation (e.g. Fig. 1d and related text).

As per claim 22, Yates discloses the method of claim 16 wherein the memory reference object includes a physical memory reference object or a logical memory reference object (col. 7:14-25 "... physical memory reference ... reference may record the event of a sequential execution flow..." and col. 17:15-20 "... memory references referring to logical address ...").

As per claim 23, this is the method version of the claimed software tool discussed above (Claim 3), wherein all claim limitations have been addressed and/or covered in cited areas as set forth above. Thus, accordingly, these claims are also anticipated by Yates.

As per claim 24, this is the method version of the claimed software tool discussed above (Claim 4), wherein all claim limitations have been addressed and/or covered in cited areas as set forth above. Thus, accordingly, these claims are also anticipated by Yates.

As per claim 25, this is the method version of the claimed software tool discussed above (Claim 6), wherein all claim limitations have been addressed and/or covered in cited areas as set forth above. Thus, accordingly, these claims are also anticipated by Yates.

As per claim 26, this is the method version of the claimed software tool discussed above (Claim 7), wherein all claim limitations have been addressed and/or covered in cited areas as set forth above. Thus, accordingly, these claims are also anticipated by Yates.

As per claim 27, this is the method version of the claimed software tool discussed above (Claim 8), wherein all claim limitations have been addressed and/or covered in cited areas as set forth above. Thus, accordingly, these claims are also anticipated by Yates.

As per claim 29, this is the method version of the claimed software tool discussed above (Claim 10), wherein all claim limitations have been addressed and/or covered in cited areas as set forth above. Thus, accordingly, these claims are also anticipated by Yates.

As per claim 30, this is the method version of the claimed software tool discussed above (Claim 11), wherein all claim limitations have been addressed and/or covered in cited areas as set forth above. Thus, accordingly, these claims are also anticipated by Yates.

As per claim 31, this is the method version of the claimed software tool discussed above (Claim 12), wherein all claim limitations have been addressed and/or covered in cited areas as set forth above. Thus, accordingly, these claims are also anticipated by Yates.

As per claim 32, this is the computer program product version of the claimed method discussed above (Claim 16), wherein all claim limitations have been addressed and/or covered in cited areas as set forth above. Thus, accordingly, these claims are also anticipated by Yates.

As per claim 33 (Currently Amended), Yates discloses a method of profiling code executing in a computer, the method comprising:

associating data addresses with memory reference objects, wherein the data addresses

have been determined from instruction instances corresponding to code execution hindrance (col.

2:15-25 "… figures out the address …"); and

aggregating the data addresses based on their associated memory reference

objects (e.g. Fig. 1c and related text), wherein the data addresses include virtual addresses in the

computer system (see also, col. 4:45-55, col.17:30-50 col.22:45-55 and col.69:40-55).

As per claim 34, this is the method version of the claimed software tool discussed above

(Claim 8), wherein all claim limitations have been addressed and/or covered in cited areas as set

forth above. Thus, accordingly, these claims are also anticipated by Yates.

As per claim 35, this is the method version of the claimed software tool discussed above

(Claim 14), wherein all claim limitations have been addressed and/or covered in cited areas as set

forth above. Thus, accordingly, these claims are also anticipated by Yates.

As per claim 36, this is another method version of the claimed method discussed above

(Claim 17), wherein all claim limitations have been addressed and/or covered in cited areas as set

forth above. Thus, accordingly, these claims are also anticipated by Yates.

As per claim 37, this is another method version of the claimed software tool discussed

above (Claim 15), wherein all claim limitations have been addressed and/or covered in cited

areas as set forth above. Thus, accordingly, these claims are also anticipated by Yates.

As per claim 39, this is another method version of the claimed software tool discussed above (Claim 11), wherein all claim limitations have been addressed and/or covered in cited areas as set forth above. Thus, accordingly, these claims are also anticipated by Yates.

As per claim 40, Yates discloses this is the computer program product version of the claimed method discussed above (Claim 33), wherein all claim limitations have been addressed and/or covered in cited areas as set forth above. Thus, accordingly, these claims are also anticipated by Yates.

As per claim 41 (Currently Amended), Yates discloses a method of profiling code in a computer system comprising:

identifying an instruction instance corresponding to a runtime event (col. 2:15-25 "… figures out the address …" and col. 7:15-40 "… execution speed … instance of instruction have an event code that leaves intact an event code previously determined…");

determining whether the instruction instance is valid (col. 6:51-59 "… profile information is recorded that records physical memory reference …" and col. 7:15-40 "… execution speed … instance of instruction have an event code that leaves intact an event code previously determined …");

decoding the instruction instance to extract at least a portion of a data address(e.g. Fig. 5b, 556 and related text)

if the instruction instance is valid (e.g. Fig. 5b, 556 and related text);

determining a memory reference object with the extracted portion of the
address (col. 7:14-25 "… physical memory reference … reference may record the event of a
sequential execution flow…" and col. 17:15-20 "… memory references referring to logical
address …); and

aggregating the data address with other addresses based at least in part on the
memory reference object, wherein the memory reference object includes virtual addresses in the
computer system (see also, col. 4:45-55, col.17:30-50 col.22:45-55 and col.69:40-55 and e.g.
Fig. 1c and related text).

As per claim 42, this is another method version of the claimed software tool discussed
above (Claim 10), wherein all claim limitations have been addressed and/or covered in cited
areas as set forth above. Thus, accordingly, these claims are also anticipated by Yates.

As per claim 43, this is another method version of the claimed software tool discussed
above (Claim 13), wherein all claim limitations have been addressed and/or covered in cited
areas as set forth above. Thus, accordingly, these claims are also anticipated by Yates.

As per claim 44, this is another method version of the claimed software tool discussed
above (Claim 14), wherein all claim limitations have been addressed and/or covered in cited
areas as set forth above. Thus, accordingly, these claims are also anticipated by Yates.

As per claim 45, this is another method version of the claimed method discussed above (Claim 21), wherein all claim limitations have been addressed and/or covered in cited areas as set forth above. Thus, accordingly, these claims are also anticipated by Yates.

As per claim 46, this is the computer program product version of the claimed method discussed above (Claim 41), wherein all claim limitations have been addressed and/or covered in cited areas as set forth above. Thus, accordingly, these claims are also anticipated by Yates.

As per claim 47 (Currently Amended), this is the computer program product version of the claimed method discussed above (Claim 41), wherein all claim limitations have been addressed and/or covered in cited areas as set forth above. Thus, accordingly, these claims are also anticipated by Yates.

As per claim 48, this is the computer program product version of the claimed method discussed above (Claim 42), wherein all claim limitations have been addressed and/or covered in cited areas as set forth above. Thus, accordingly, these claims are also anticipated by Yates.

As per claim 49, this is the computer program product version of the claimed method discussed above (Claim 45), wherein all claim limitations have been addressed and/or covered in cited areas as set forth above. Thus, accordingly, these claims are also anticipated by Yates.

As per claim 50, this is the computer program product version of the claimed software tool discussed above (Claim 13), wherein all claim limitations have been addressed and/or covered in cited areas as set forth above. Thus, accordingly, these claims are also anticipated by Yates.

As per claim 53, this is the computer program product version of the claimed software tool discussed above (Claim 6), wherein all claim limitations have been addressed and/or covered in cited areas as set forth above. Thus, accordingly, these claims are also anticipated by Yates.

As per claim 54, this is computer program product version of the claimed software tool discussed above (Claim 7), wherein all claim limitations have been addressed and/or covered in cited areas as set forth above. Thus, accordingly, these claims are also anticipated by Yates.

As per claim 55 (Currently Amended), this is the apparatus version of the claimed method discussed above (Claim 41), wherein all claim limitations have been addressed and/or covered in cited areas as set forth above. Thus, accordingly, these claims are also anticipated by Yates.

As per claim 56, this is the apparatus version of the claimed method discussed above (Claim 42), wherein all claim limitations have been addressed and/or covered in cited areas as set forth above. Thus, accordingly, these claims are also anticipated by Yates.

   As per claim 58, Yates discloses the apparatus of claim 56 wherein the processor includes

event condition counters (col. 2:15-25 "… counters will indicate …").

                                    Conclusion

8.   Any inquiry concerning this communication or earlier communications from the

examiner should be directed to ISAAC T. TECKLU whose telephone number is (571) 272-7957.

The examiner can normally be reached on M-TH 9:300A - 8:00P.

   If attempts to reach the examiner by telephone are unsuccessful, the examiner's

supervisor, Tuan Q. Dam can be reached on (571) 272-3695. The fax phone number for the

organization where this application or proceeding is assigned is 571-273-8300.

   Information regarding the status of an application may be obtained from the Patent

Application Information Retrieval (PAIR) system. Status information for published applications

may be obtained from either Private PAIR or Public PAIR. Status information for unpublished

applications is available through Private PAIR only. For more information about the PAIR

system, see http://pair-direct.uspto.gov. Should you have questions on access to the Private PAIR

system, contact the Electronic Business Center (EBC) at 866-217-9197 (toll-free). If you would

like assistance from a USPTO Customer Service Representative or access to the automated

information system, call 800-786-9199 (IN USA OR CANADA) or 571-272-1000.

/Isaac T Tecklu/   /Tuan Q. Dam/
Examiner, Art Unit 2192   Supervisory Patent Examiner, Art Unit 2192